BASIC PROGRAMMING

UNCLASSIFIED

RISK ASSESSMENT:

LOW

SAFETY CONSIDERATIONS:

ELECTRICAL SHOCK HAZARD
ELECTROSTATIC DISCHARGE

ENVIRONMENTAL CONSIDERATION

NONE

Terminal Learning Objective

ACTION:

Construct a computer program.

CONDITIONS:

Given a classroom environment, a Nida 500 trainer, the 8086 instruction set and official OMEMS reference material.

STANDARDS:

Correctly construct a computer program that will run error free.

Enabling Learning Objective

ACTION:

Construct a flow chart of a computer program.

CONDITIONS:

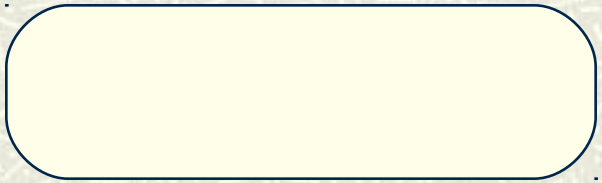
Given a classroom environment, the 8086 instruction set and official OMEMS reference material.

STANDARDS:

Correctly flow chart a computer program that will run error free.

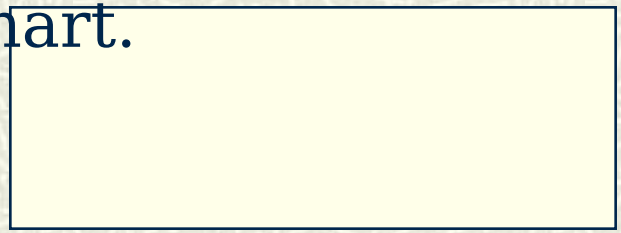
FLOW CHART

- A flow chart is a graphical or logical system, consisting of an interconnected series of symbols, each indicating a certain operation or action.



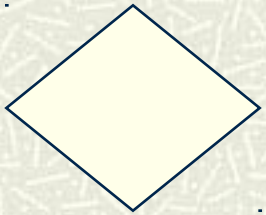
TERMINAL SYMBOL

Terminal symbols (start, stop, delay, halt) are used to designate the beginning, end and temporary halt on a flow chart.



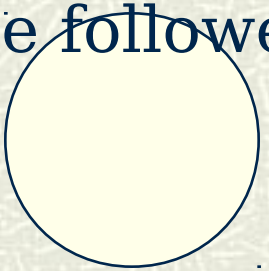
PROCESS SYMBOL

Process symbols are used to designate an arithmetical, logical or data transfer operations on the flow chart.



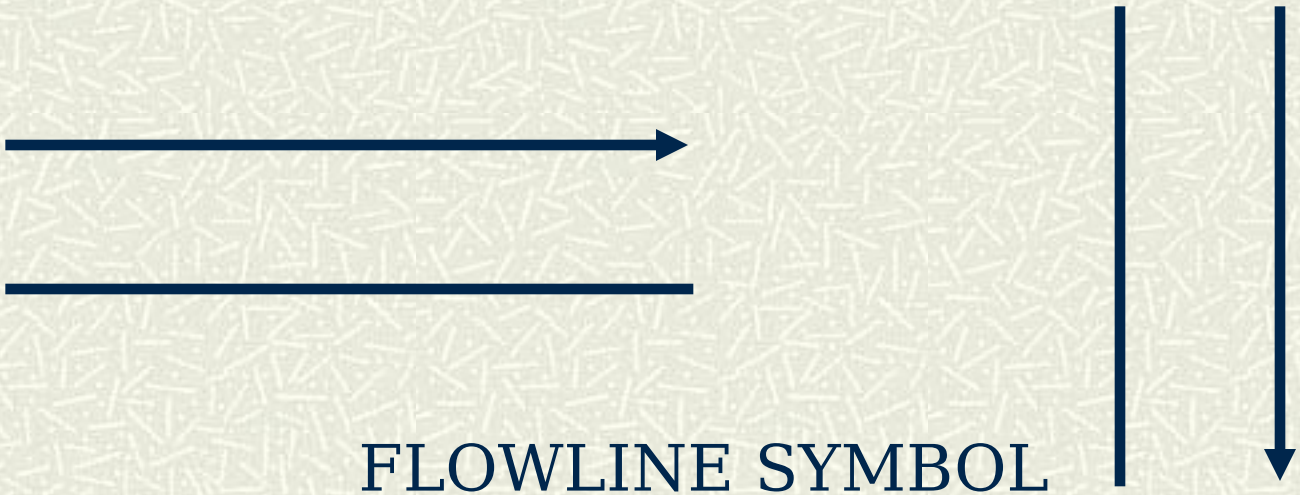
DECISION SYMBOL

Decision symbols are used to designate logical choices. They determine which of a number of alternate paths is to be followed.



CONNECTOR SYMBOL

Connector symbols are used to designate the junction of flow lines between different parts of a flow chart.



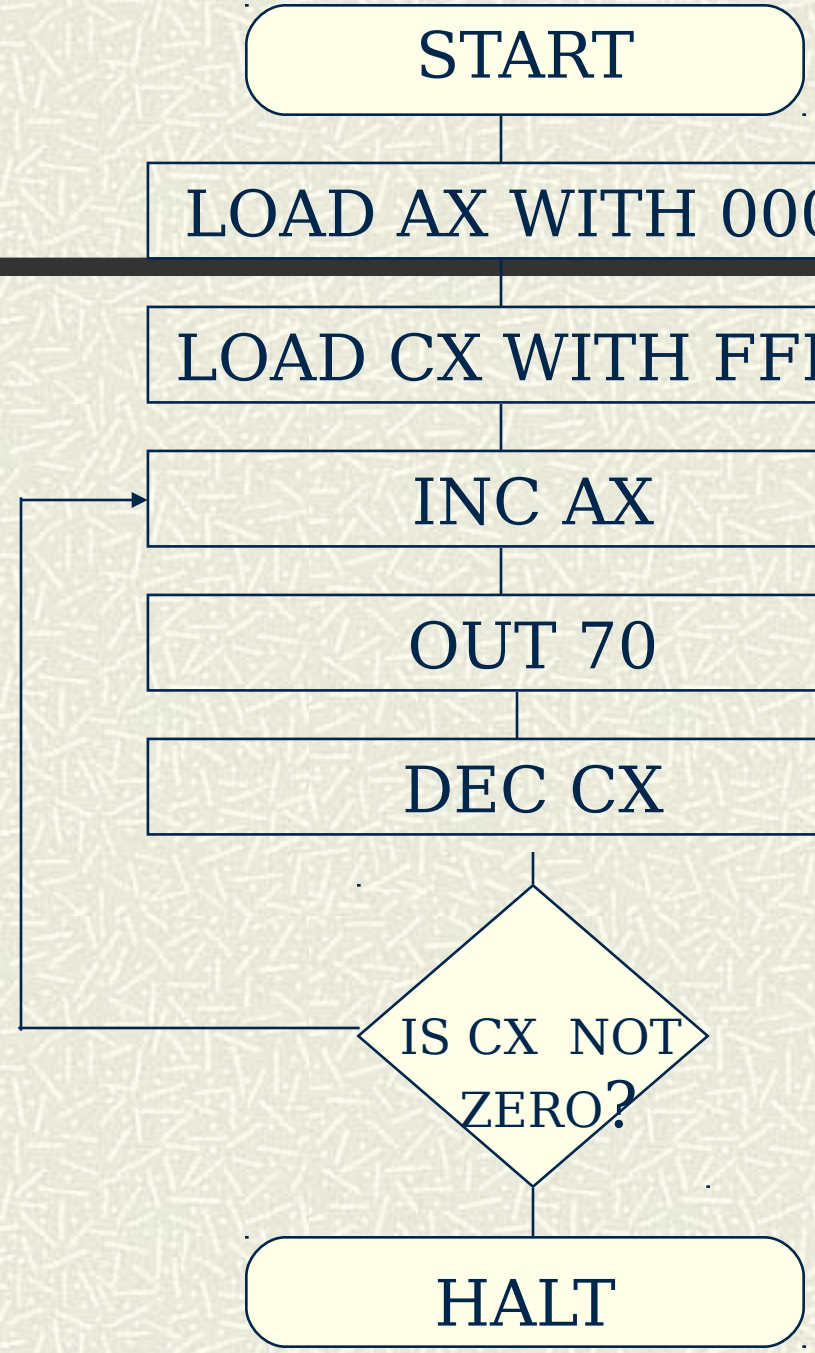
FLOWLINE SYMBOL

Flowline symbols are used to designate the direction of flow between symbols. Normal direction of flow is from left to right and top to bottom. When increased clarity is desired, open arrowheads can be placed on normal direction flowlines.

WRITING A PROGRAM

- The first step of writing a program is to develop a flow chart of the problem to be solved, or the operation that needs to take place.
- Construct a flow chart that will verify the operation of the parallel user I/O port.

CF351K07-1



Note: If CX is not zero the program jumps back to the INC AX instruction.

PROGRAM

- A list of instructions that are arranged in a specified order to control the operation of a microprocessor system or computer. The program tells the machine what to do on a step-by-step basis.

Enabling Learning Objective

ACTION:

Explain the use of the 8086 instruction set.

CONDITIONS:

Given a classroom environment, a Nida 500 trainer, the 8086 instruction set and official OMEMS reference material.

STANDARDS:

Correctly construct a computer program that will run error free.

INSTRUCTION TYPES

DATA TRANSFER INSTRUCTIONS

Transfer data between registers or between memory and registers.

ARITHMETIC INSTRUCTIONS

Add, subtract, increment or decrement data in registers or memory.

LOGICAL INSTRUCTIONS

AND, OR, XOR, NOT, compare and complement data in memory or registers.

CONTROL TRANSFER INSTRUCTIONS

Conditional and unconditional jump instructions

Enabling Learning Objective

ACTION:

Write a program correctly.

CONDITIONS:

Given a classroom environment, a Nida 500 trainer, the 8086 instruction set and official OMEMS reference material.

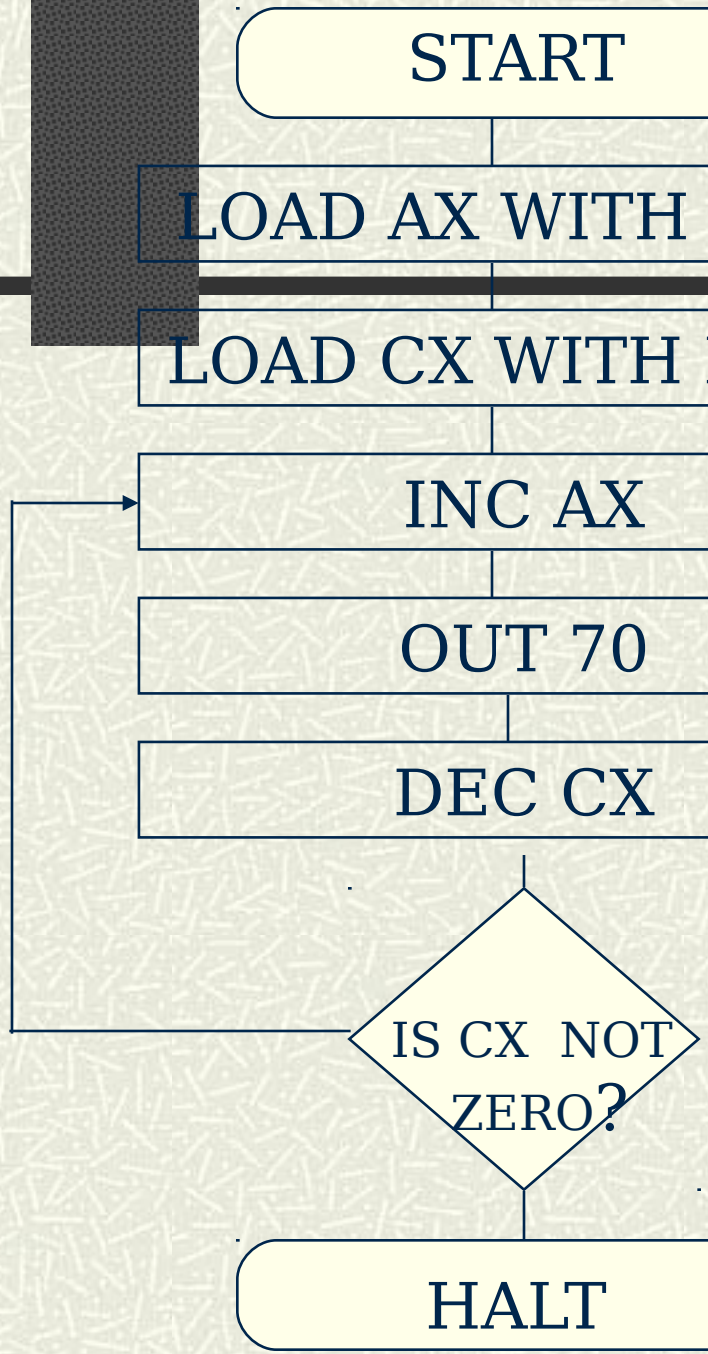
STANDARDS:

Correctly construct a computer program that will run error free.

SAMPLE PROGRAM

- WRITE A PROGRAM THAT WILL VERIFY THE OPERATION OF THE NIDA 500 TRAINER OUTPUT PORT.
- Starting address will be 1000h. Load 0000h into the AX, load FFFFh into the CX, increment AX and output the contents to the User I/O port. Decrement the CX register, jump back to the increment AX instruction if CX is not zero, terminate the program if CX is zero.
- Construct the flow chart and then write the program.

CF351K07-17



ADDR	CODE	MNEMONIC
1000	B8	MVI AX, 0000
1001	00	
1002	00	
1003	B9	MVI CX, FFFF
1004	FF	
1005	FF	
1006	40	INC AX
1007	E7	OUT AX
1008	70	
1009	49	DEC CX
100A	75	JNZ FA
100B	FA	
100C	CD	HALT
100D	20	

DEBUGGING PROGRAMS

- ERROR
- MISTAKE
- MALFUNCTION

Enabling Learning Objective

ACTION:

Verify computer program operations.

CONDITIONS:

Given a classroom environment, a Nida 500 trainer, the 8086 instruction set and official OMEMS reference material.

STANDARDS:

Correctly construct a computer program that will run error free.

PE #1

- Starting address will be 1000h. You are to put the value of 0006h into location 5050h/5051h. Put the value 0009h into the BX register. Add these numbers and store the results into location 6000h, and then halt the program

PE #2

- Starting address will be 3000h. You are to put a value of AAA9h into the AX register. Increment the AX register by one. Put a value of 5556h into the BX register. Decrement the BX register by one. Subtract the contents of the BX register from the contents of the AX register and send the results to the User I/O port display, and then halt the program.

PE #3

- Using Register command load AX with 2155h. Load BX with 5521h. Using the Examine command load address 6000h with FFh and 6001h with FFh. Starting with address 5000h, load SI with 6000h. Compare AX with 5521h. Compare BX with 5521h. Compare memory location 6000h with 5521h. Halt the program.

PE #4

- Starting with address 1000h, load AL with 00h. Load CX with FFFFh. Increment the AL and output it's contents to the user I/O port. Decrement the CX. Jump back to increment AL instruction if CX is not zero. Halt the program if CX is zero.

PE #5

- Starting with address 3000h, load AL with D5h. Output AL contents to user I/O port. Complement AL. Output AL to user I/O port. Increment AL. Output AL to user I/O port. Halt program.

SUMMARY

- FLOW CHARTS
- 8086 INSTRUCTION SET
- DEBUGGING